First Hit Fwd Refs

☐ Generate Collection Print

L2: Entry 252 of 279 File: USPT Jun 27, 1995

US-PAT-NO: 5428810
DOCUMENT-IDENTIFIER: US 5428810 A

TITLE: Allocation of resources of a pipelined processor by clock phase for parallel execution of dependent processes

DATE-ISSUED: June 27, 1995

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Barkans; Anthony C.	Fort Collins	CO		
Swanson; Roger	Fort Collins	CO		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Hewlett-Packard Company	Palo Alto	CA			02

APPL-NO: 08/ 220948 [PALM]
DATE FILED: March 31, 1994

PARENT-CASE:
CROSS REFERENCE TO RELATED APPLICATION This is a continuation of application Ser. No. 07/669,824 filed on Mar. 15, 1991, now abandoned.

INT-CL: [06] G06 F 13/00, G06 F 9/38

US-CL-ISSUED: 395/800; 395/550, 364/230, 364/231.4, 364/231.8, 364/271.2, 364/DIG.1

US-CL-CURRENT: 712/42; 709/400, 712/217, 713/400

FIELD-OF-SEARCH: 395/800, 395/775, 395/725, 395/425, 395/325, 395/575, 395/550, 364/230, 364/231.4, 364/231.8, 364/271.2, 364/DIG.1

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected Search ALL Clear

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	4354227	October 1982	Hays, Jr. et al.	395/725
☐	4481578	November 1984	Hughes et al.	395/425
☐	4594655	June 1986	Hao et al.	395/775

☐	4695943	September 1987	Keeley et al.	395/425
☐	4789927	December 1988	Hannah	364/200
☐	5060145	October 1991	Scheuneman et al.	395/425

OTHER PUBLICATIONS

Patterson and Hennessy, Computer Architecture: A Quantitative Approach, Morgan Kaufmann Publishers, San Mateo, Calif., 1990, pp. 250-349.
A. C. Barkans, "High Speed High Quality Antialiased Vector Generation", Computer Graphics, vol. 24, No. 4, Aug. 1990, pp. 319-326.

ART-UNIT: 232

PRIMARY-EXAMINER: Bowler; Alyssa H.

ASSISTANT-EXAMINER: Harrity; John

ABSTRACT:

A technique of processing pipeline commands in parallel so as to minimize pipeline stalls. This is accomplished in accordance with the invention without need for the complex resource allocation techniques of the prior art by arbitrating access to critical pipeline resources on the phase of the system clock. For example, one control process may access the critical pipeline resource only during an even phase of the system clock, while a second control process may access the critical pipeline resource only during the odd phase of the clock. These processes may run at the same time if the pipelined instructions being executed by each process have no data dependencies since structural hazards are effectively eliminated by time-sharing the data buses on the respective phases of the system clock. The benefits of dynamically scheduled pipelined systems may thus be obtained without the complex scoreboarding and other scheduling algorithms used in the prior art to prevent pipeline hazards.

7 Claims, 14 Drawing figures

First Hit Fwd Refs

☐ Generate Collection Print

L2: Entry 252 of 279 File: USPT Jun 27, 1995

DOCUMENT-IDENTIFIER: US 5428810 A
TITLE: Allocation of resources of a pipelined processor by clock phase for parallel execution of dependent processes

Brief Summary Text (16):
For example, scoreboarding is a sophisticated prior art technique for dynamically scheduling around hazards by allowing instructions to execute out of order when there are sufficient resources and no data dependencies. In particular, a scoreboard is used to separate the process of issuing an instruction into two parts, namely, checking the structural hazards and waiting for the absence of a data hazard. Structural hazards can be checked when an instruction is issued; however, if the instructions are to begin execution as soon their data operands are available, the pipeline will have to perform out of order execution. Scoreboarding makes this possible.

Brief Summary Text (17):
The goal of a scoreboard is to maintain an execution rate of one instruction per clock cycle when there are no structural hazards by executing an instruction as early as possible. Thus, when an instruction at the front of an input queue is stalled, other instructions can be issued and executed if they do not depend on any active or stalled instruction. The scoreboard takes full responsibility for instruction issue and execution, including all hazard detection. Taking advantage of out of order execution requires multiple instructions to be in their execution stage simultaneously. This can be achieved with either multiple functional units or with pipelined functional units. Accordingly, a scoreboard acts as a means for resource allocation which checks for hazards and then allocates the instructions accordingly.

Brief Summary Text (18):
FIG. 1 illustrates a pipelined processing system utilizing a scoreboard. As shown, a plurality of registers 100 are provided which are accessed by respective pipelined processing circuits 102-108 to perform pipelined processing functions on the data stored therein. A scoreboard 110 is provided for receiving every instruction and constructing a picture of the data dependencies of the respective instructions. This picture is then used by scoreboard 110 to determine when an input instruction can read its operands and begin execution. If scoreboard 110 decides the instruction cannot execute immediately, it monitors every change in the hardware and decides when the instruction can execute. Scoreboard 110 also controls when an instruction can write its result into its destination register. Thus, all hazard detection and resolution is centralized in the scoreboard 110.

Brief Summary Text (19):
Scoreboard 110 also controls the instruction progression from one step to the next by communicating with the functional units 102-108. However, since there is only a limited number of source operands and result buses to the registers 100, scoreboard 110 must guarantee that the functional units allowed to proceed do not require more than the number of data busses available. In other words, the data busses are treated by the scoreboard as resources which must be allocated. This added complexity often causes the scoreboard 110 to have about as much logic as one of the functional units and, on average, about four times as many data busses as would

be required if the pipeline only executed instructions in order. Such complexity is undesirable, and it is desired that relatively simple techniques be developed for efficient pipelined processing. In particular, an alternative to the use of scoreboards for special purpose pipelined processing systems is desired.

Brief Summary Text (25):
The above-mentioned shortcomings in the prior art have been met in accordance with the present invention by developing a technique for dynamic scheduling of a pipelined processor for parallel execution of multiple processes. The technique of the invention is quite straightforward and eliminates the need for complex resource allocation algorithms of the type used in prior art pipelined processing systems of the type described above. In particular, the present invention allocates critical resources shared by multiple processes by synchronizing the processes to the respective phases of the system clock. As used herein, a critical resource is a resource of the pipeline which must be accessed by a process to continue its command execution and to prevent pipeline stalls. In other words, certain input commands are given access to shared critical resources only during particular phases of the system clock. By so arbitrating access to shared critical resources based upon the phase of the system clock, instructions need not be executed in sequential order so long as there are no data dependencies between the respective instructions. Moreover, such an arrangement effectively implies a scoreboard of the type used in the prior art from the hardware configuration of the system, whereby resource allocation is inherent in the design. In particular, there is no need to check for a structural hazard or wait for the absence of a data hazard since the respective processes share critical resources in a time-division multiplexed manner. The present invention thus resolves, without adding undue complexity, the resource allocation problems which have heretofore limited the performance of pipelined processing systems.

Drawing Description Text (3):
FIG. 1 illustrates a prior art pipelined processing system in which a scoreboard is used to allow pipeline instructions to execute out of order when there are sufficient resources and no data dependencies between respective instructions.

Detailed Description Text (18):
Accordingly, in the preferred embodiment of the invention, when input PLA 302 informs pipeline control circuit 304 that valid data has been received, control process 1 is started to handle the input commands. A source file listing of a PLA which performs the functions of control process 1 is attached below as an Appendix. Control process 1 is assigned an even (or odd) phase of the system clock and hence only controls access to critical resources such as command register 306 during the assigned clock phase. During the unassigned clock phase, access to the critical resources is not permitted. However, when a rendering command is recognized by control process 1, control process 2 is started and assigned an opposing phase of the clock for accessing critical resources. Control process 2 then reads the next instruction from the command register 306 to determine if the next instruction is one that can be processed while control process 1 controls the polygon render. The aforementioned Appendix also sets forth a preferred embodiment of PLA code for implementing control process 2. Control process 2 is shown in the Appendix as "machine side door", and as shown therein it is a very small piece of code, much simpler than the aforementioned compiler-scheduling code or scoreboard hardware.

Current US Original Classification (1):
712/42

Current US Cross Reference Classification (2):
712/217